

Fig. 1A

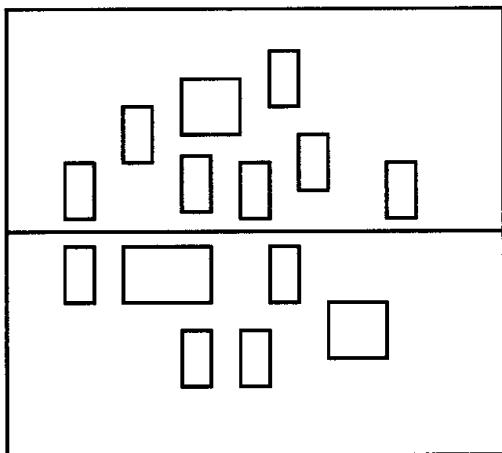


Fig. 1B

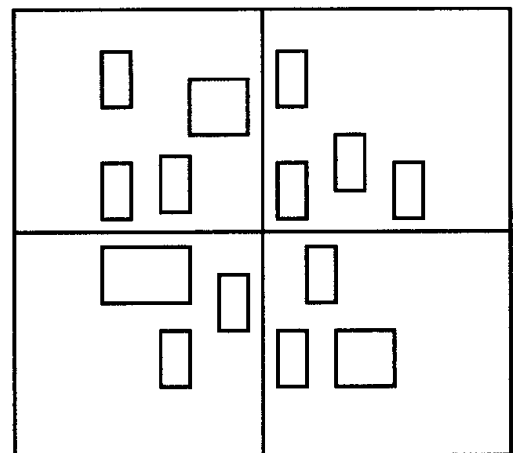
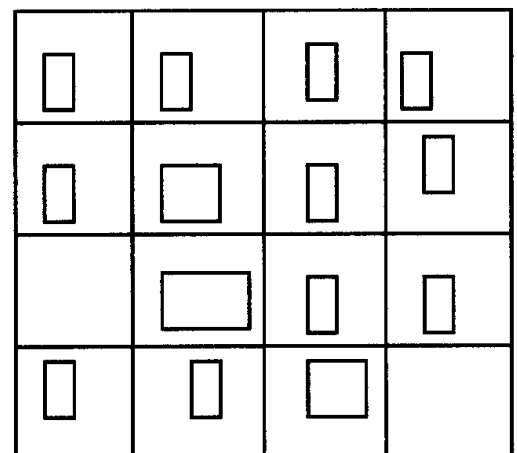
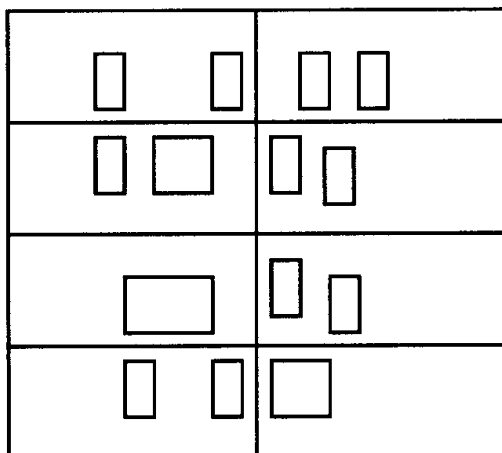


Fig. 1C



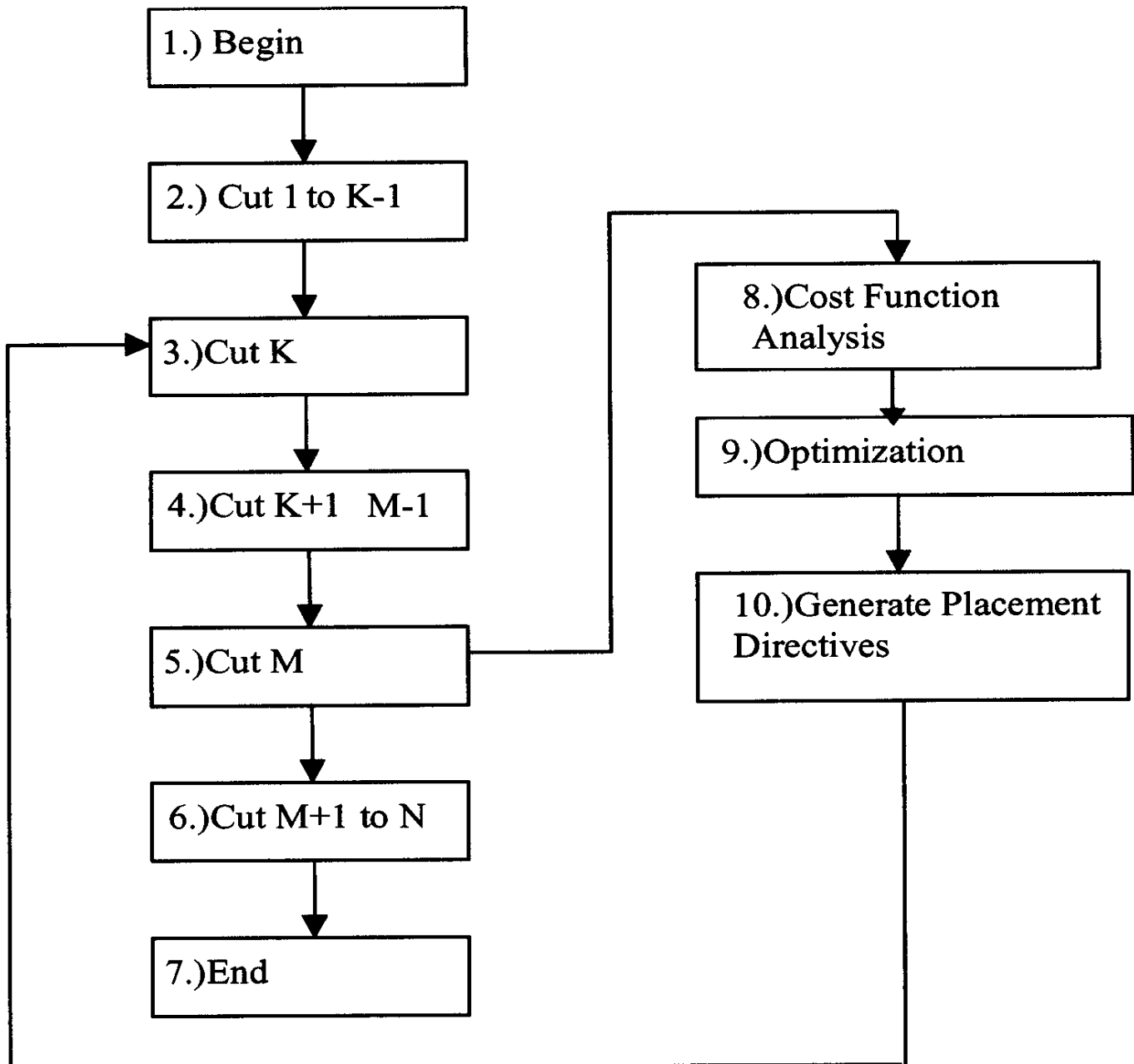


Figure 2

```
1.      placement_flow(N)
      {
2.      K = user_specified_value;
3.      Look_ahead_factor = user_specified_value;
4.      M = N * look_ahead_factor;

5.      do_M_cuts();
      /* Next 3 lines: perform optimization (timing, power, congestion, signal integrity,
      etc) */
6.      do_cost_function_analysis
7.      do_cost_function_optimization
8.      generate_placement_directives

9.      restore_placement_to_cut(K);
10.     do_remaining_cuts_K_to_N();
      }
```

Figure 3

```

1.      Placement_flow(N)
      {
2.      K = user_specified_value1;
3.      Look_ahead_factor = user_specified_value2;
4.      Forward_increment = user_specified_value3;
5.      C = user_specified_value4;
6.      K = C - forward_increment;
7.      Do_K_cuts(); /* cut = K */

8.      M = N * look_ahead_factor;
9.      While ((iterations < max_iterations) & (K < N)); /* termination criteria */
      {
10.     restore_placement_for_cut_K(); /* cut = K */
11.     do_cuts_K_through_M();

12.     /* perform optimization (timing, power, congestion, signal integrity, etc) */

13.     do_cost_function_analysis()
14.     do_cost_function_optimization()
15.     generate_placement_directives()
16.     K = K + Forward_Increment;
17.     If(K > N) K = N;
18.     M = M + Forward_Increment;
19.     If ( M > N) M = N;
      }
20.     if (cut != N)
      {
21.         K = cut;
22.         Do_cuts_K_through_N();
      }
      }

```

Figure 4

```

1.      placement_flow(expansion_rate)
      {
2.          do_M_cuts();
3.          run_global_router();
4.          congestion_map = congestion_estimation();
5.          expand_cells (congestion_map, expansion_rate);
6.          insert_blockage(congestion_map);
7.          undo_cuts_K_to_M();
8.          do_remaining_cuts_K_to_N();
      }
9.      expand_cells (congestion_map, expansion_rate)
      {
10.         for each movable 1byn cell in a vertical congested grid
            {
11.                 base_expansion = base_expansion + pin_number / cell_width
12.                 congested_cell_area = congested_cell_area + cell_width
            }
13.         target_expansion = congested_cell_area * expansion_rate
14.         K = target_expansion / base_expansion;
15.         for each movable 1byn cell in a vertical congested grid
            {
16.                 new_cell_width = cell_width + pin_number / cell_width * K
17.                 assign new_cell_width to the cell
            }
      }
18.      insert_blockage(congestion_map)
      {
19.          identify horizontal congested grids
20.          create rectangle regions by merging neighboring horizontal congested
            grids for each region
            {
21.                 N = blockages density, which is depend on congestion degree
22.                 create blockage every N number of circuit row with one
                    circuit row high and region wide
            }
      }

```

Figure 5

FIS920030335US1

Haoxing Ren, et al.

6/8

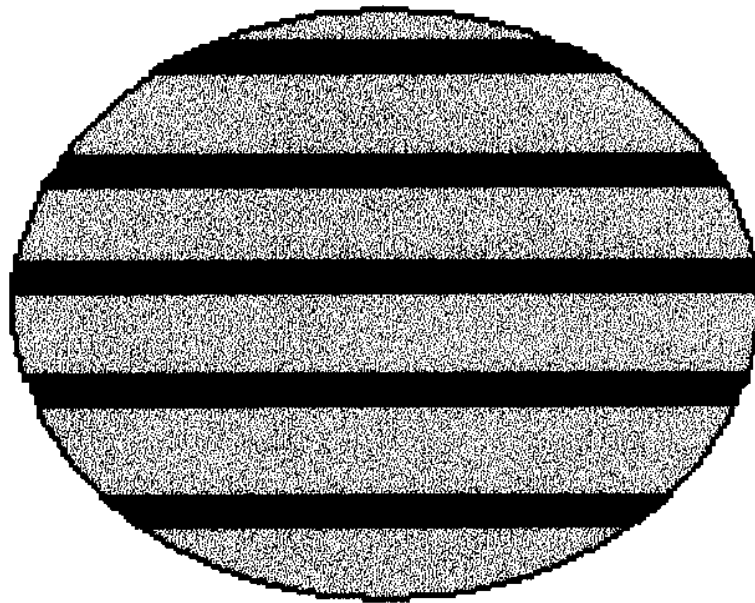


Figure 6

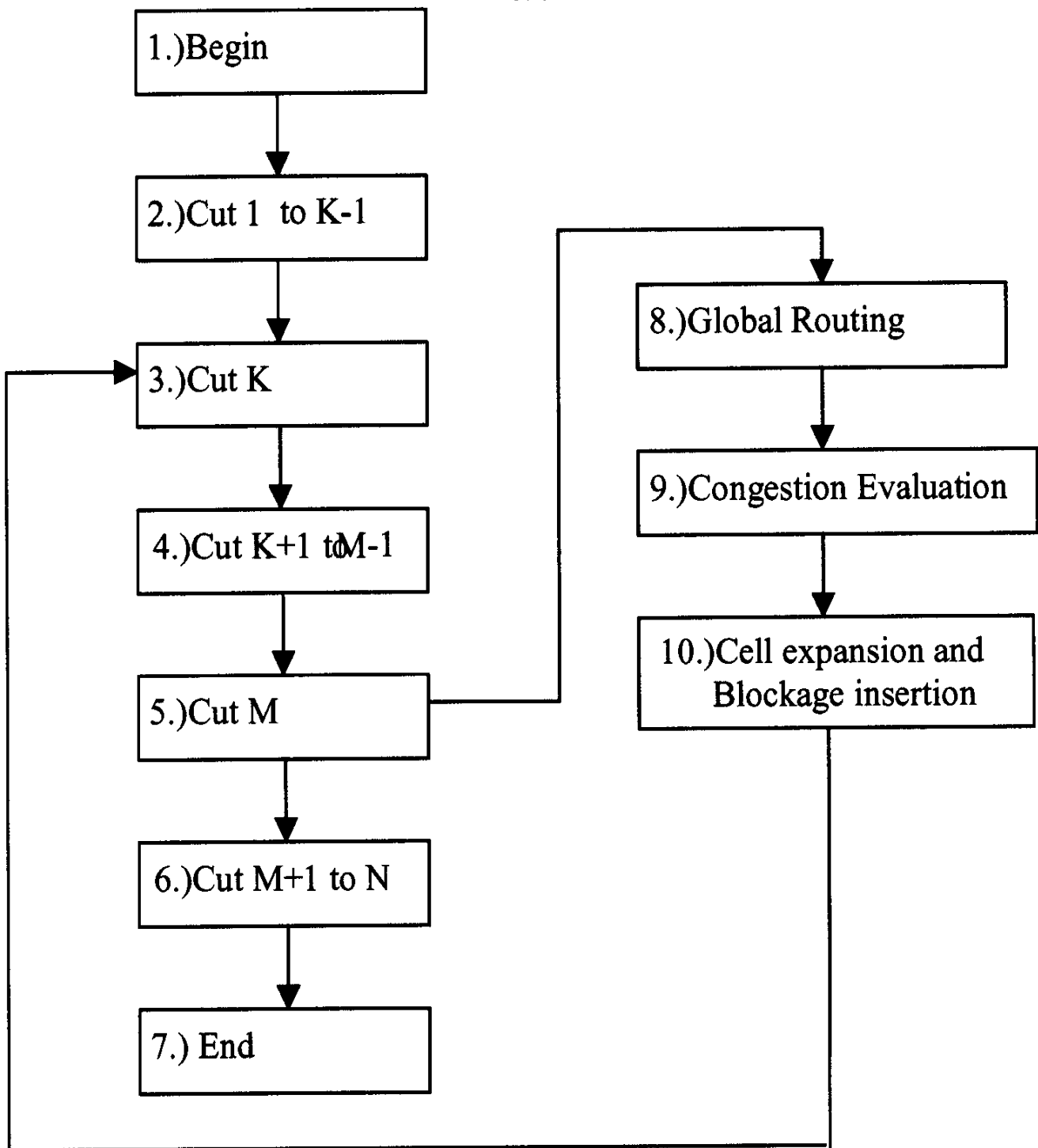


Figure 7

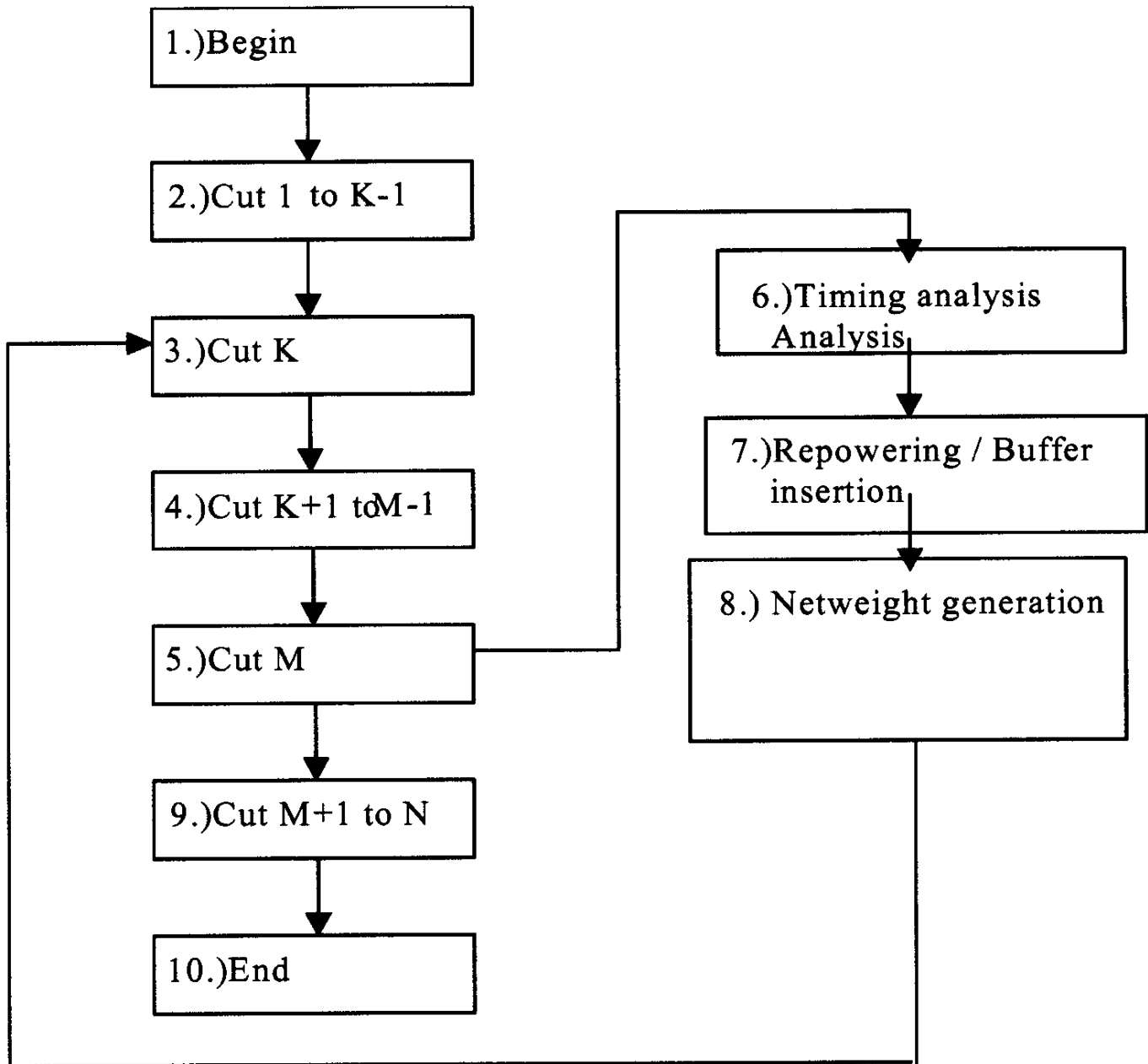


Figure 8